

**Proiect RIW**

**Motor de căutare având ca sursă de  
documente un set local de fișiere  
folosind MongoDB**

**Student: Condriea Stefan-Catalin**

**Grupa: 1410A**

**Profesor: Alexandru Archip**

Proiectul de față are ca scop găsierea unor fișiere care conțin o anumită(sau anumite) informație, având ca suport un set de fișiere suficient de mare.

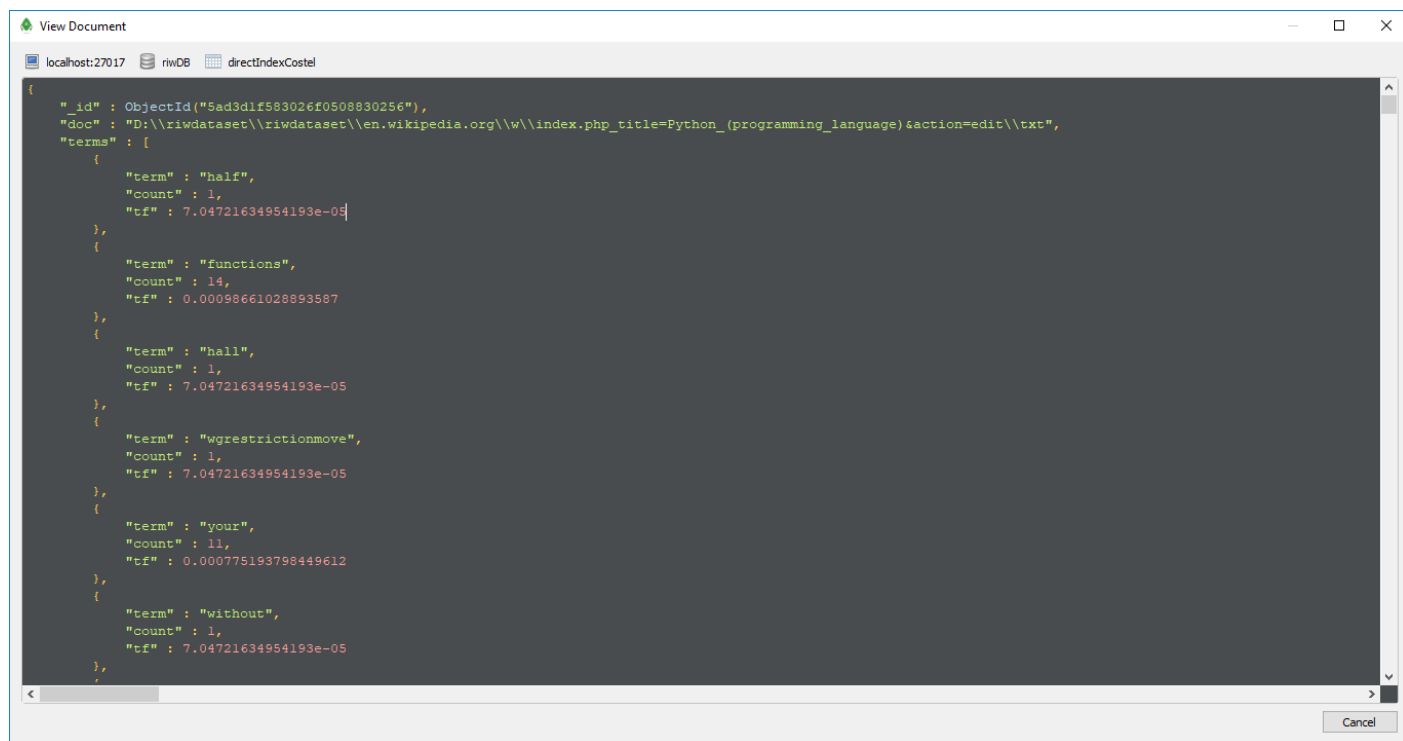
Pașii implicați pentru realizarea acestui sunt:

### 1. Crearea indexului direct

Preluarea dintr-un fișier („stopWords.txt”) a unor stopWord-uri din limba engleză și, pe baza acestora crearea unei liste (mai exact un hashMap) cu acestea și, în mod analog a unei liste de excepții.

Se parcurgere recursiv setului de fișiere și directoare, se preia textului din fiecare fișier, se filtrează cuvintele prin lista de stopWord-uri și excepții, și se stochează într-un HashMap, avand ca și cheie cuvântul propriu-zis, iar ca valoare, numărul de apariții a cuvântului în documentul respectiv.

Pe baza acestui hashMap, se creează un **bson.Document** ca in figura de mai jos,



```
{
  "_id" : ObjectId("5ad3d1f583026f0508830256"),
  "doc" : "D:\\riwdataset\\riwdataset\\en.wikipedia.org\\w\\index.php_title=Python_(programming_language)&action=edit\\txt",
  "terms" : [
    {
      "term" : "half",
      "count" : 1,
      "tf" : 7.04721634954193e-05
    },
    {
      "term" : "functions",
      "count" : 14,
      "tf" : 0.00098661028893587
    },
    {
      "term" : "hall",
      "count" : 1,
      "tf" : 7.04721634954193e-05
    },
    {
      "term" : "wgrestrictionmove",
      "count" : 1,
      "tf" : 7.04721634954193e-05
    },
    {
      "term" : "your",
      "count" : 11,
      "tf" : 0.000775193798449612
    },
    {
      "term" : "without",
      "count" : 1,
      "tf" : 7.04721634954193e-05
    }
  ]
}
```

unde term este cuvântul propriu zis, count-numarul de apariții al acestuia, iar tf(term frequency) este frecența de apariție în document.

Acesta este stocat într-o colecție dintr-o baza de date mongo.

Analog se face pentru toate documentele.

Clasa folosită este **DirectIndex** și metoda **createIndex(inputFolderPath);**

## 2. Crearea indexului invers

Dacă la indexul direct există documentul și o listă de cuvinte corespunzătoare acestuia, la indexul invers este cuvântul și lista de documente în care se află. Indexul invers se creează pe baza indexului direct.

!!! Indexul invers se creează în memorie și apoi se încarcă în mongo.

În acest timp, pentru fiecare cuvânt calculăm și idf-ul, (modul de calcul al acestui se găsește în Cursul 3, slide 15);

## 3. Crearea vectorului corespunzător fiecărui document și calculul modulului

Având tf-ul și idf-ul calculați de la pașii anteriori, se revine în colecția de date a indexului direct și pentru fiecare document se creează un vector corespunzător, și se calculează modulul acestuia.

Acesta va fi adăugat ca și câmp nou în fiecare document din indexul direct.

Clasa folosită este **InverseIndex** și metoda **createInverseIndexLocal ()**;

## 4. Modulul de căutare

Pentru query-ul introdus, se creează un document similar celor din indexul direct și se calculează similaritatea Cosinus între fiecare document și query.

Cele mai “aproprate” documente vor fi afișate în browser.

Pentru aceasta se folosește clasa **Search** prin metoda **getList(Document query)**.

Proiect:

Proiect de tip Maven, java pe partea de server,

Framework: spring

Server de aplicație embedded: Apache

Pentru testarea acestui proiect este nevoie de instalat baza de date Mongo, și java (versiune 8 sau mai mult).

Pentru rulare se rulează în cmd în interiorul proiectului(unde este fișierul mvnw.cmd) comanda **mvnw spring-boot:run** și se așteaptă câteva secunde. La prima rulare se vor descărca dependențele de care are nevoie într-un repository local. După ce serverul este pornit se face și indexarea(directa+inversă) și crearea vectorului corespunzator documentelor.

De asemenea, trebuie deschis serverul pentru mongo (host="localhost", port=27107).

După aceasta, se deschide în browser pagina: <http://localhost:8080/> si apare o pagina simpă unde se introduce un query și (după enter) se așteaptă câteva secunde o listă de documente.

Baza de date se numește RIWBrowser.

Colecția indexului invers se numește **directIndex**, iar cea pentru indexul invers – **inverseIndex**. De reținut că în cazul în care există în baza de date colecții cu acest nume și conțin cel puțin un document, indexarea nu se mai face.